CSE 492 Software Systems Engineering

“Are CORBA Objects Reliable in Real-Time Systems?”

by

06 December 1999

Introduction

Object oriented technologies have made a tremendous impact in the development of computer systems in recent years. One popular trend is the use of the Common Object Request Architecture (CORBA). According to Dr. Alexander Malinowski, Department of Electrical Engineering University of Wyoming, "CORBA emerged as one of possible solutions to data exchange among distributed applications." This communication protocol for interaction between heterogeneous systems has generated a demand from members of the telecommunications and defense community. Both require systems that meet real time specifications and are considered to be "mission critical." This paper attempts to discuss the development of this technology and research its reliability in embedded systems.

Defining CORBA

CORBA provides the standards that specify a common architecture for building distributed object systems, common services for use in such systems, and protocols for interoperability among implementations. The Object Management Group (OMG) produced these standards, a group consisting have over 600 software vendors, developers and end users. Object Request Brokers (ORBs) are middleware products that implement CORBA standards. CORBA's goal is to allow systems with differing hardware components, operating systems, and programming languages to communicate regardless of their location on a network. Included in the standards is a platform independent language the Interface Definition Language (IDL) which allows a client application to invoke the methods of a server object in a very seamless fashion.

Defining Real-Time Systems

What is a Real-Time system? "A Real-Time system is one in which the correctness of the computations not only depends upon the logical correctness of the computations but also upon the time at which the result is produced. If the timing constraints of the system are not met, system failure is said to have occurred.", Donald Gillies, Software Engineering, University of British Columbia, Vancouver, B.C. Canada. Such systems usually demand high reliability and dependability because of safety implications and the potential lost of life. Therefore it is essential to meet timing constraints for these systems to behave in the appropriate manner.

CORBA Real Time Issue

OMG's 1991 and 1994 adoption of CORBA standards, did not specifically address the use of CORBA in Real Time and fault-tolerant settings, still many embedded systems were implemented with CORBA extensions. But problems with CORBA supporting Real-Time application remained unsolved as late as October 1998. Developers were required to over-architect systems to ensure data flow with no interruption and to determine their quality of service. Developing predictable ORBs and providing control to ORB operations were not specified in the standards. Additionally, ORB's handling of task priority information was based on proprietary decisions. Solutions or lack of solutions to solve such problems defeated the purpose of CORBA or required complex design work by the developer.

Building Real-Time distributed systems with predictability and schedulability is challenging enough. Real-Time operation systems provide a predictable thread

scheduling but other components such as processor, buses and networks required their on schedule. What is required in real time CORBA is the ability to schedule when components run and prioritize their activities over the network, says OMG officials. ORBs used under the 1991 and 1994 standards had implementation options as to how to make CORBA work. One option was to specify quality of service properties in the IDL, which required knowledge of all network nodes at design time. This was fine for closed system architectures but the industry was moving towards open systems architectures. Another option was to set these properties during the connection of the client and server at run time, which became a challenging design issue. One implementation problem was the choice of passing task priority information from the client to the server, or assigning static priorities to the server objects. TCP/IP is the dominant transport protocol used with CORBA. There were demands for systems to use proprietary transports other than TCP/IP to meet application Real-Time requirements but this jeopardized their ability to talk to other nodes on the network. The use of ORBs in embedded systems desired additional standards to ensure their reliability in these critical systems.

Studies

In December of 1997, Andreus Polze of Humbolt University of Berlin lead a team in studying the use of CORBA in Real-Time settings focusing on problems in the manufacturing domain. Using the Software Engineering Institute (SEI) simplex architecture, the first model problem examined “the use of ORBs as an interconnection mechanism between hard Real-Time systems”. Secondly, they explored the use of the National Institute of Standards (NIST) Real-Time Control System (RCS) architecture to

model a problem of ORBs as a Real-Time communication mechanism within a Real-Time system.

In the first problem the CORBA extended simplex architecture is designed to support and tolerate errors introduced by new or upgraded components. A synchronized inverted pendulum application is executed to demonstrate the control of a physical device through fault-tolerate software. Here, CORBA is not responsible for achieving Real-Time and quality-of-services requirements. After examining results, the following concerns were voiced: 1) Although not in this case, communication latency could be a problem using a non-Real-Time gateway between real time applications 2) Sharing computer platforms between the Real-Time application and the CORBA methods disrupted the applications behavior because of network loading of the RTOS when numerous CORBA methods were invoked.

The second problem studied the use of CORBA Communication as a substitute for shared-memory communication inside the NIST motion controller soft Real-Time application. It used the RCS to control a machine tool. Observations showed latency problems with the CORBA substitute. Because of this, key components read invalid data that cause the motor of the tool to behave inappropriately. The test revealed the following:

1) Simply parting an ORB to the Real-Time platform is not sufficient to integrate CORBA and Real-Time programming.

2) The performance of the system could improve with tuning.

Polze report concluded that CORBA could be used in certain Real-Time application settings in which many of CORBA's shortcomings would have to be addressed.

The Defense Information Infrastructure Common Operating Environment Integrated Product Team (DII-COE IPT) presented a Real-Time ORB Trade Study in August 1999. ORB products studied included Hardpack by Lockheed Martin, ORB express by Objective Interface and TAO by Washington University. The test executed six primary IDC operations with a single client thread in four different scenarios, client/server on the same machine, client and server machines separated by 10 MB Ethernet with a zero and 70 millisecond delays between invocations. Analysis of the results revealed all three performed better than non-Real-Time ORB products in which ORB express out performed its competitors. ORB express V.2.0.1 product was benchmark with 2,700 two-way operations per second by Lawrence Livermore Labs.

HardPack has been sponsored as part of the Real-Time DII COE. Originally developed to support time-critical military applications, HardPack implement CORBA services as well as custom Real-Time and fault-tolerant extensions. Lockheed addressed many of the issues associated with CORBA for embedded systems. HardPack provides Real-Time predictability and streamlined low latency, it manages resources allocations and preserves priorities. HardPack was also used in the modernization effort for the Airborne Warning and Control System (AWACS) computer infrastructure.

Real-Time CORBA Adopted

Finally in the spring of 1999, the OMG adopted standards for Real-Time ORBs (CORBA 3.0). The primary goal of the specification was to support developers in meeting Real-Time systems requirements. Real-Time CORBA is defined as an extension to the existing CORBA. Developers now have ORBs implemented with the ability to do

"end-to-end predictability" a key requirement of fault-tolerant systems. Thread priorities between client and server will be respected for resolving resource contention. Developers can set the latencies of operation invocation and bound the duration of thread priority inversions. Real-Time CORBA system will require four key components:

1) The scheduling mechanisms in the OS
2) The Real-Time ORB
3) The communication transport
4) The application (s)

The important point to make here is that Real-Time ORB must still rely on the underlying operating system and the application to determine predictability.

Other specifications addressing problem areas such as resource management and compatibility are incorporated in this version. Real-Time CORBA provides control over threadpools by assigning objects threads and their priorities and transport connections. Using the standard CORBA Internet Inter-ORB Protocol (IIOP), two Real-Time ORBs can be interoperable and IIOP will allow vendors to bridge between Real-Time ORBs by mapping their Real-Time IOP onto the IIOP. Real-Time CORBA components can interwork with CORBA components but with predictability and latency issues that must be addressed and CORBA applications can be ported to Real-Time ORBs.

Conclusion

CORBA has become a very popular technology within the IT community. The benefits of using CORBA for solving distributed system problems, providing cost effective flexible and portable systems and promoting reuse and etc are attractive to Real-Time system developers. Initially the specifications for CORBA did not address its use in Real-Time settings. Inspite of this, embedded systems were implemented using CORBA extensions. Research has shown that fault-tolerant systems using these extensions are reliable in certain settings. Many vendors specifically designed and develop their products to address Real-Time CORBA issues and satisfied the needs of this niche market. Based on my research I feel confident that CORBA can be implemented in Real-Time systems. The newly adopted Real-Time CORBA specification has addressed many of the earlier issues concerning hard Real-Time systems. Although I believe there will be modifications and additions in later version but for now CORBA is reliable.

References

1. Polze, A. (1997). A Study in the Use of CORBA in Real-Time Settings: Model Problems for the Manufacturing Domain. Software Engineering Institute
2. The Object Management Group's home page is www.omg.org.
3. Murphy, N. (1998). Introduction to CORBA for Embedded Systems. Embedded Systems Programming, October 1998
4. Lockheed Martin Federal Systems: www.owego.com/hardpack
5. Malinowski, A. (1998). CORBA – the New Technology for Industrial and Telecommunication Network-Distributed Applications. New Technologies, http://sant.bradley.edu/~ienews/98_2/corba.htm
6. Garon, J and Herscher, S. (1997) Embedded CORBA. Expersoft Developer Resources, http://www.expersoft.com/Resources/Wpapers/embed_corba.htm
7. Gonsalves, A. (1998) CORBA Goes Real-Time. PC Week, 1998
8. Objective Interface Systems, Inc. (1998). Real-Time and Embedded Corba Objective Interface Systems, Inc.
9. Comp.realtime: Frequently Asked Questions (FAQs) URL is http.faqs.org/faqs/realtime-computing/faq/
10. The DII COE Real-Time ORB Trade Study webpage is http://www.ois.com/technical/rt_diicoe_ipt.html, (1999)
11. Orfali R., Harkey D., and Edwards J., (1999) Client/Server Survival Guide. John Wiley & Sons, Inc.